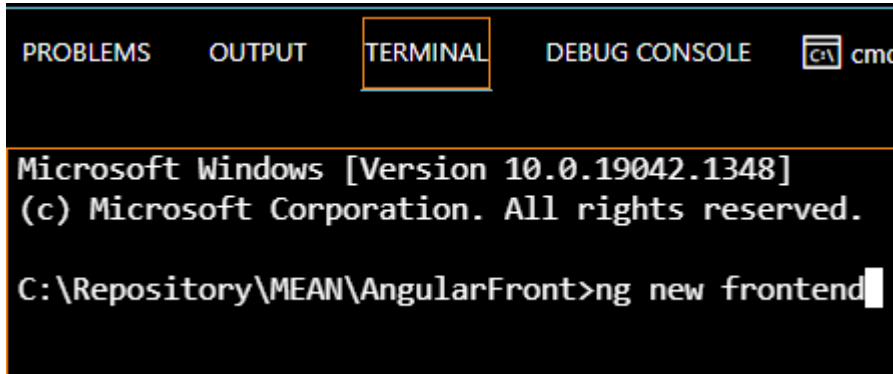


1. Creación del proyecto AngularFront.

✓ **ANGULARFRONT**

2. Creación de folder frontend en la terminal de comandos con el comando **ng new frontend**.

Se realiza de esta manera pues así indicamos que esta será una carpeta Angular.

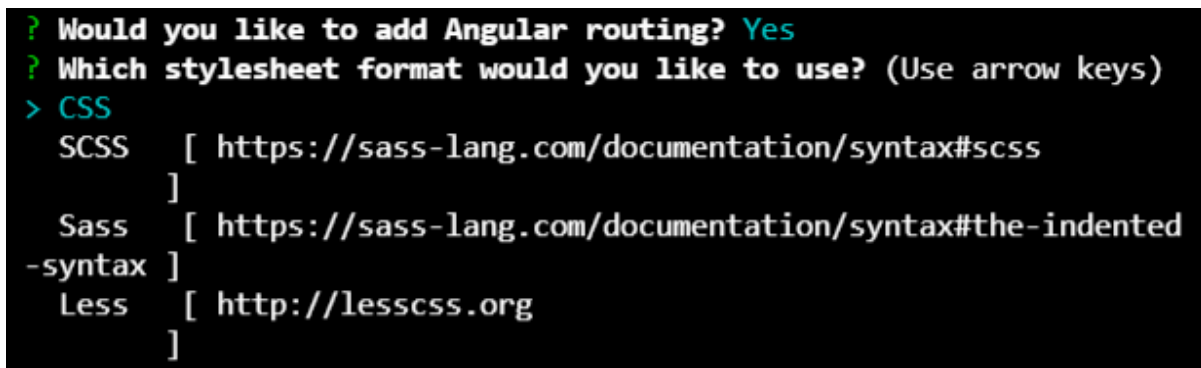


```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE cmd
Microsoft Windows [Version 10.0.19042.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Repository\MEAN\AngularFront>ng new frontend
```

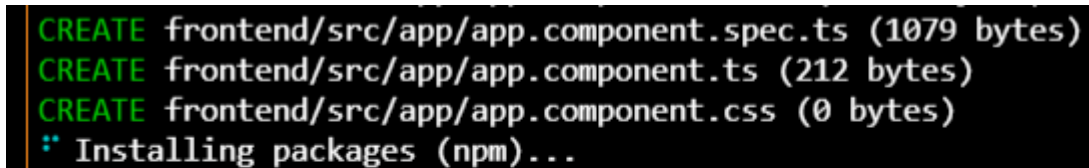
2.1 Iniciamos la configuración de nuestro folder FrontEnd en Angular con dos preguntas:

- a. Nos informa que si queremos instalar routing en nuestro proyecto para que por medio de rutas podemos consumir los diferentes componentes.
- b. Que tipo de diseño queremos para nuestro proyecto, y seleccionamos CSS



```
? Would you like to add Angular routing? Yes
? Which stylesheet format would you like to use? (Use arrow keys)
> CSS
  SCSS [ https://sass-lang.com/documentation/syntax#scss
    ]
  Sass [ https://sass-lang.com/documentation/syntax#the-indented
-syntax ]
  Less [ http://lesscss.org
    ]
```

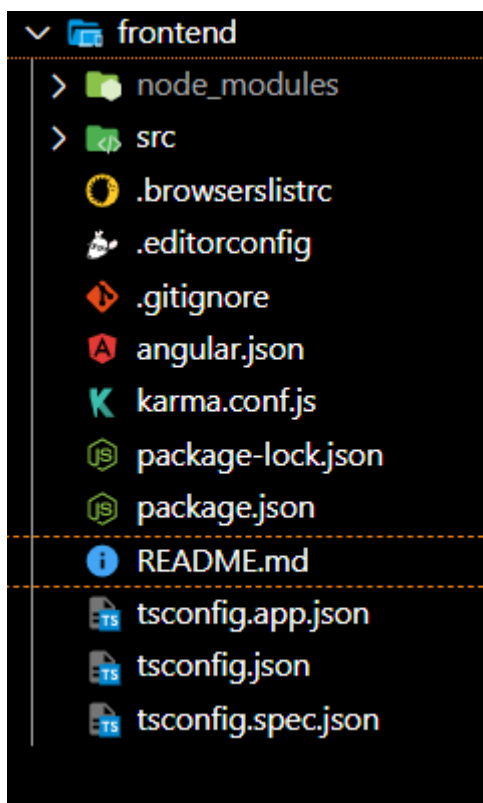
Una vez confirmadas estas dos preguntas continuamos con la instalación de paquetes.



```
CREATE frontend/src/app/app.component.spec.ts (1079 bytes)
CREATE frontend/src/app/app.component.ts (212 bytes)
CREATE frontend/src/app/app.component.css (0 bytes)
* Installing packages (npm)...
```

```
warning: LF will be replaced by CRLF in tsconfig.spec.json.  
The file will have its original line endings in your working direc  
tory  
    Successfully initialized git.  
  
C:\Repository\MEAN\AngularFront>
```

Después de esto, están entonces creadas todas las configuraciones y librerías necesarias para el funcionamiento de angular en nuestro proyecto front.



2.2 Continuado podemos validar que angular esté funcionando y que todo esté ok iniciando el servidor con el comando en terminal: **ng serve**.

```
C:\Repository\MEAN\AngularFront\frontend>ng serve  
Generating browser application bundles (phase: setup)...
```

Así debe ver en caso de que sí esté todo OK. Recordando que funcionará en nuestro buscador a través del protocolo `http://localhost:` en el puerto 4200/

```
** Angular Live Development Server is listening on localhost:4200,  
open your browser on http://localhost:4200/ **  
  
✓ Compiled successfully.
```

3. Iniciamos con la creación de componentes principales de nuestro home con el comando **ng g c** . **ng** = Angular **g** = generate **c** = component. Todo esto basándome de cierta forma en el ejemplo de biblioteca realizado anteriormente.

```
C:\Repository\MEAN\AngularFront\frontend>ng g c home/header
CREATE src/app/home/header/header.component.html (21 bytes)
CREATE src/app/home/header/header.component.spec.ts (626 bytes)
CREATE src/app/home/header/header.component.ts (275 bytes)
CREATE src/app/home/header/header.component.css (0 bytes)
UPDATE src/app/app.module.ts (480 bytes)

C:\Repository\MEAN\AngularFront\frontend>ng g c home/footer
CREATE src/app/home/footer/footer.component.html (21 bytes)
CREATE src/app/home/footer/footer.component.spec.ts (626 bytes)
CREATE src/app/home/footer/footer.component.ts (275 bytes)
CREATE src/app/home/footer/footer.component.css (0 bytes)
UPDATE src/app/app.module.ts (567 bytes)

C:\Repository\MEAN\AngularFront\frontend>ng g c home/slider-formularios
CREATE src/app/home/slider-formularios/slider-formularios.component.html (33 bytes)
CREATE src/app/home/slider-formularios/slider-formularios.component.spec.ts (704 bytes)
CREATE src/app/home/slider-formularios/slider-formularios.component.ts (322 bytes)
CREATE src/app/home/slider-formularios/slider-formularios.component.css (0 bytes)
UPDATE src/app/app.module.ts (700 bytes)
```

Se agregaron los componentes en la carpeta home/ header - footer y slider-formularios. El último pensado como un componente en donde se visualicen formularios como si fueran imágenes y sean seleccionados según se requiera, registro o login por ejemplo.

4. Organizamos los componentes principales en el folder app en el file app.component.html

```
1 <app-header></app-header>
2 <app-slider-formularios></app-slider-formularios>
3 <router-outlet></router-outlet>
4 <app-footer></app-footer>
```

5. Luego de organizar estos componentes pasamos a otros que nos ayudaran con los procesos o tareas planteados para la app: Registrar libros, buscar libros, lista de libros, updatebook.

```
C:\Repository\MEAN\AngularFront\frontend>ng g c task/register-book
CREATE src/app/task/register-book/register-book.component.html (28 bytes)
CREATE src/app/task/register-book/register-book.component.spec.ts (669 bytes)
CREATE src/app/task/register-book/register-book.component.ts (302 bytes)
CREATE src/app/task/register-book/register-book.component.css (0 bytes)
UPDATE src/app/app.module.ts (813 bytes)

C:\Repository\MEAN\AngularFront\frontend>ng g c task/search-book
CREATE src/app/task/search-book/search-book.component.html (26 bytes)
CREATE src/app/task/search-book/search-book.component.spec.ts (655 bytes)
CREATE src/app/task/search-book/search-book.component.ts (294 bytes)
CREATE src/app/task/search-book/search-book.component.css (0 bytes)
UPDATE src/app/app.module.ts (918 bytes)

C:\Repository\MEAN\AngularFront\frontend>ng g c task/listar-book
CREATE src/app/task/listar-book/listar-book.component.html (26 bytes)
CREATE src/app/task/listar-book/listar-book.component.spec.ts (655 bytes)
CREATE src/app/task/listar-book/listar-book.component.ts (294 bytes)
CREATE src/app/task/listar-book/listar-book.component.css (0 bytes)
UPDATE src/app/app.module.ts (1023 bytes)
```

6. Creación de servicios a consumir, (tomando el ejemplo relacionado al inicio).

```
C:\Repository\MEAN\AngularFront\frontend>ng g s services/books
CREATE src/app/services/books.service.spec.ts (352 bytes)
CREATE src/app/services/books.service.ts (134 bytes)
```

6.1 Los servicios no se agregan de forma automática en app.module.ts por lo tanto debemos hacerlo de forma manual.



```
import {BooksService} from "../services/books.service";
```

<pre>], providers: [], bootstrap: [AppComponent]</pre>	<pre>providers: [BooksService], bootstrap: [AppComponent]</pre>
--	---

7. Creación de guard para que por ejemplo en este caso solo las peticiones desde url registradas/logeadas puedan utilizar nuestra app. Es la proteccion en el front.

7.1 Se despliegan varias opciones para cumplir con la descripción del guard, CanActivate es la opción más general y nos permite decidir.

```
C:\Repository\MEAN\AngularFront\frontend>ng g guard guard/auth
? Which interfaces would you like to implement? (Press <space> to select, <a> to toggle all, <i> to invert selection, and <enter> to proceed)
>(*) CanActivate
  ( ) CanActivateChild
  ( ) CanDeactivate
  ( ) CanLoad
```

7.2 Importamos en appmodules.ts.

```
import {AuthGuard} from './guard/auth.guard';
```

```
providers: [BooksService, AuthGuard],
```

8. Continuamos el folder de variables de entorno, para crear una la cual nos ayude conectándonos con la variable de entorno de nuestro servidor en backend.

```
APP_URL: 'http://localhost:3001/api/'
```

9. Vamos a el file `app-routing` la cual nos ayuda con la navegación por lo tanto agregaremos todos los componentes.

En donde `path:` no ayuda declarando inicialmente que vamos a declarar una ruta local.

La primera está vacía debido a que es la primera en salir es decir el archivo `app.component`.

```
path: '',
```

Continuando con los demás componentes de usuarios.

```
path: 'listar-book',  
component: ListarBookComponent,  
},  
{  
  path: 'register-book',  
  component: RegisterBookComponent,  
},  
{  
  path: 'search-book',  
  component: SearchBookComponent,  
},  
{  
  path: 'update-book',  
  component: UpdateBookComponent,
```